

The read/write protocol complex is collapsible^{*}

Fernando Benavides^{1,2} and Sergio Rajsbaum¹

¹ Instituto de Matemáticas, Universidad Nacional Autónoma de México,
Ciudad Universitaria, D.F. 04510 México

rajsbaum@im.unam.mx

² Departamento de Matemáticas y Estadística, Universidad de Nariño,
San Juan de Pasto, Colombia
fandresbenavides@gmail.com

Abstract. The celebrated *asynchronous computability theorem* provides a characterization of the class of decision tasks that can be solved in a wait-free manner by asynchronous processes that communicate by writing and taking atomic snapshots of a shared memory. Several variations of the model have been proposed (immediate snapshots and iterated immediate snapshots), all equivalent for wait-free solution of decision tasks, in spite of the fact that the protocol complexes that arise from the different models are structurally distinct. The topological and combinatorial properties of these snapshot protocol complexes have been studied in detail, providing explanations for why the asynchronous computability theorem holds in all the models.

In reality concurrent systems do not provide processes with snapshot operations. Instead, snapshots are implemented (by a wait-free protocol) using operations that write and read individual shared memory locations. Thus, read/write protocols are also computationally equivalent to snapshot protocols. However, the structure of the read/write protocol complex has not been studied. In this paper we show that the read/write iterated protocol complex is collapsible (and hence contractible). Furthermore, we show that a distributed protocol that wait-free implements atomic snapshots in effect is performing the collapses.

1 Introduction

A *decision task* is the distributed equivalent of a function, where each process knows only part of the input, and after communicating with the other processes, each process computes part of the output. For example, in the *k-set agreement* task processes have to agree on at most k of their input values; when $k = 1$ we get the *consensus* task [8].

A central concern in distributed computability is studying which tasks are solvable in a distributed computing model, as determined by the type

^{*} A short version to appear in the Springer LNCS proceedings of LATIN 2016. Partially supported by UNAM-PAPIIT grant.

of communication mechanism available and the reliability of the processes. Early on it was shown that consensus is not solvable even if only one process can fail by crashing, when asynchronous processes communicate by message passing [8] or even by writing and reading a shared memory [22]. A graph theoretic characterization of the tasks solvable in the presence of at most one process failure appeared soon after [3].

The *asynchronous computability theorem* [15] exposed that moving from tolerating one process failure, to any number of process failures, yields a characterization of the class of decision tasks that can be solved in a wait-free manner by asynchronous processes based on simplicial complexes, which are higher dimensional versions of graphs. In particular, n -set agreement is not wait-free solvable, even for $n + 1$ processes [4,15,24].

Computability theory through combinatorial topology has evolved to encompass arbitrary malicious failures, synchronous and partially synchronous processes, and various communication mechanisms [13]. Still, the original wait-free model of the asynchronous computability theorem, where crash-prone processes that communicate wait-free by writing and reading a shared memory is fundamental. For instance, the question of solvability in other models (e.g. f crash failures), can in many cases be reduced to the question of wait-free solvability [7,14].

More specifically, in the *AS model* of [13] each process can write its own location of the shared-memory, and it is able to read the whole shared memory in one atomic step, called a *snapshot*. The characterization is based on the *protocol complex*, which is a geometric representation of the various possible executions of a protocol. Simpler variations of this model have been considered. In the *immediate snapshot* (IS) version [2,4,24], processes can execute a combined write-snapshot operation. The *iterated immediate snapshot* (IIS) model [6] is even simpler to analyze, and can be extended (IRIS) to analyze partially synchronous models [23]. Processes communicate by accessing a sequence of shared arrays, through immediate snapshot operations, one such operation in each array. The success of the entire approach hinges on the fact that the topology of the protocol complex of a model determines critical information about the solvability of the task and, if solvable, about the complexity of solution [17].

All these snapshot models, AS, IS, IIS and IRIS can solve exactly the same set of tasks. However, the protocol complexes that arise from the different models are structurally distinct. The combinatorial topology properties of these complexes have been studied in detail, providing insights for why some tasks are solvable and others are not in a model.

Results In reality concurrent systems do not provide processes with snapshot operations. Instead, snapshots are implemented (by a wait-free protocol) using operations that write and read individual shared memory locations [1]. Thus, read/write protocols are also computationally equivalent to snapshot protocols. However, the structure of the read/write protocol complex has not been studied. Our results are the following.

1. The one-round read/write protocol complex is collapsible to the IS protocol, i.e. to a chromatic subdivision of the input complex. The collapses can be performed simultaneously in entire orbits of the natural symmetric group action. We use ideas from [21], together with distributed computing techniques of partial orders.
2. Furthermore, the distributed protocol that wait-free implements immediate snapshots of [5,9] in effect is performing the collapses.
3. Finally, also the multi-round iterated read/write complex is collapsible. We use ideas from [10], together with carrier maps e.g. [13].

All omitted proofs are in the Appendix.

Related work The one-round immediate snapshot protocol complex is the simplest, with an elegant combinatorial representation; it is a chromatic subdivision of the input complex [13,19], and so is the (multi-round) IIS protocol [6]. The multi-round (single shared memory array) IS protocol complex is harder to analyze, combinatorially it can be shown to be a pseudomanifold [2]. IS and IIS protocols are homeomorphic to the input complex. An AS protocol complex is not generally homeomorphic to the underlying input complex, but it is homotopy equivalent to it [12]. The span of [15] provides an homotopy equivalence of the (multi-round) AS protocol complex to the input complex [12], clarifying the basis of the obstruction method [11] for detecting impossibility of solution of tasks.

Later on stronger results were proved, about the collapsibility of the protocol complex. The one-round IS protocol complex is collapsible [20] and homeomorphic to closed balls. The structure of the AS is more complicated, it was known to be contractible [12,13], and then shown to be collapsible (one-round) to the IS complex [21]. The IIS (multi-round) version was shown to be collapsible too [10].

There are several wait-free implementations of atomic snapshots starting with [1], but we are aware of only two algorithms that implement immediate snapshots; the original of [5], and its recursive version [9].

2 Preliminaries

2.1 Distributed computing model

The basic model we consider is the one-round *read/write* model (WR), e.g. [16]. It consists of $n + 1$ processes denoted by the numbers $[n] = \{0, 1, \dots, n\}$. A process is a deterministic (possibly infinite) state machine. Processes communicate through a shared memory array $\text{mem}[0 \dots n]$ which consists of $n + 1$ single-writer/multi-reader atomic registers. Each process accesses the shared memory by invoking the atomic operations $\text{write}(x)$ or $\text{read}(j)$, $0 \leq j \leq n$. The $\text{write}(x)$ operation is used by process i to write value x to register i , and process i can invoke $\text{read}(j)$ to read register $\text{mem}[j]$, for any $0 \leq j \leq n$. Each process i has an input value, which may be its own id i . In its first operation, process i writes its input to $\text{mem}[i]$, then it reads each of the $n + 1$ registers, in an arbitrary order. Such a sequence of operations, consisting of a write followed by all the reads is abbreviated by $\text{WScan}(x)$.

An *execution* consists of an interleaving of the operations of the processes, and we assume any interleaving of the operations is a possible execution. We may also consider an execution where only a subset of processes participate, consisting of an interleaving of the operations of those processes. These assumptions represent a wait-free model where any number of processes may fail by crashing.

In more detail, an execution is described as a set of atomic operations together with the irreflexive and transitive partial order given by: op precedes op' if op was completed before op' . If op does not precede op' and viceversa, the operations are called *concurrent*. The set of values read in an execution α by process i is called the *local view* of i which is denoted by $\text{view}(i, \alpha)$. It consists of pairs (j, v) , indicating that the value v was read from the j -th register. The set of all local views in the execution α is called the *view* of α and it is denoted by $\text{view}(\alpha)$. Let \mathcal{E} be a set of executions of the WR model. Consider the equivalence relation on \mathcal{E} given by: $\alpha \sim \alpha'$ if $\text{view}(\alpha) = \text{view}(\alpha')$. Notice that for every execution α there exists an equivalent *sequential* execution α' with no concurrent operations. In other words, if op and op' are concurrent operations in α we can suppose that op was executed immediately before op' without modifying any views. Thus, we often consider only sequential executions α , consisting of a linear order on the set of all write and read operations.

Two other models can be derived from the WR model. In the *iterated* WR model, processes communicate through a sequence of arrays. They all go through the sequence of arrays $\text{mem}_0, \text{mem}_1 \dots$ in the same order,

and in the r -th round, they access the r -th array, mem_r , exactly as in the one-round version of the WR model. Namely, process i executes one write to $\text{mem}_r[i]$ and then reads one by one all entries j , $\text{mem}_r[j]$, in arbitrary order. In the *non-iterated, multi-round* version of the WR model, there is only one array mem , but processes can execute several rounds of writing and then reading one by one the entries of the array. The *immediate snapshot* model (IS) [4,24], consists of a subset of executions of the WR one round model. Namely, all the executions where the operations can be organized in *concurrency classes*, each one consisting a set of writes by the set of processes participating in the concurrency class, followed by a read to all registers by each of these processes. See Section 3.1.

2.2 Algorithm IS

Consider the recursive algorithm IS of [9] for the iterated WR model, presented in Figure 1. Processes go through a series of disjoint shared memory arrays $\text{mem}_0, \text{mem}_1, \dots, \text{mem}_n$. Each array mem_k is accessed by process i invoking $\text{WScan}(i)$ in the recursive call $\text{IS}(n+1-k)$. Process i executes $\text{WScan}(i)$ (line (1)), by performing first $\text{write}(i)$, followed by $\text{read}(j)$ for each $j \in [n]$, in an arbitrary order. The set of values read (each one with its location) is what the invocation of $\text{WScan}(i)$ returns. In line (2) the process checks if view contains $n+1-k$ id's, else $\text{IS}(n-k)$ is again invoked on the next shared memory in line (3). It is important to note that in each recursive call $\text{IS}(n+1-k)$ at least one process returns with $|\text{view}| = n+1-k$, given that $n+1-k$ processes invoked IS. For example, in the first recursive call $\text{IS}(n+1)$ the last process to write reads $n+1$ values and terminates the algorithm.

Algorithm $\text{IS}(n+1)$
 (1) $\text{view} \leftarrow \text{WScan}(i)$
 (2) **if** $|\text{view}| = n+1$ **then** return view
 (3) **else** return $\text{IS}(n)$.

Fig. 1. Code for process i

Every execution of the IS protocol can be represented by a finite sequence $\alpha = \alpha_0, \alpha_1, \dots, \alpha_l$ with α_k an execution of the WR one round model where every process that takes a step in α_k invokes the recursive call with $\text{IS}(n+1-k)$. Since at least one process terminates the algorithm the length $l(\alpha) = l+1$ is at most $n+1$. The last returned local view in

execution α for process i is denoted $view(i, \alpha)$, and the set of all local views is denoted by $view(\alpha)$.

Denote by \mathcal{E}_l the set of views of all executions α with $l(\alpha) = l+1$. Then $\mathcal{E}_n \subseteq \dots \subseteq \mathcal{E}_0$. In particular, \mathcal{E}_0 corresponds to the views of executions of the one round WR of Section 2.1. Also, \mathcal{E}_n corresponds to the views of the immediate snapshot model, see Theorem 1 of [9].

3 Definition and properties of the protocol complex

Here we define the protocol complex of the write/read model and other models, which arise from the sets \mathcal{E}_i mentioned in the previous section.

3.1 Additional properties about executions

Recall from Section 2.1 that an execution can be seen as a linear order on the set of write and read operations. For a subset $I \subseteq [n]$ let

$$\mathcal{O}_I = \{w_i, r_i(j) : i \in I, j \in [n]\}.$$

with $I = \mathcal{O}_i = \emptyset$. A *wr-execution on I* is a pair $\alpha = (\mathcal{O}_I, \rightarrow_\alpha)$ with \rightarrow_α a linear order on \mathcal{O}_I such that $w_i \rightarrow_\alpha r_i(j)$ for all $j \in [n]$. The set I is called the *Id* set of α which is denoted by $\text{Id}(\alpha)$. Hence the view of i is $view(i, \alpha) = \{j \in I : w_j \rightarrow_\alpha r_i(j)\}$ and the view of α is $view(\alpha) = \{(i, view(i, \alpha)) : i \in I\}$. Note the chain $w_i \rightarrow_\alpha r_i(j_0) \rightarrow_\alpha \dots \rightarrow_\alpha r_i(j_n)$ represents the invoking of WScan by the process i in the *wr-execution* α . Consider a *wr-execution* α and suppose that the order in which the process i reads the array $\text{mem}[0 \dots n]$ is given by $r_i(j_0) \rightarrow_\alpha \dots \rightarrow_\alpha r_i(j_n)$. If every write operation w_k satisfies $w_k \rightarrow_\alpha r_i(j_0)$ or $r_i(j_n) \rightarrow_\alpha w_k$ then $view(i, \alpha)$ corresponds to an atomic snapshot.

As a consequence, every execution in the snapshot model and immediate snapshot model corresponds to an execution in the write/read model. For instance in the *wr-execution*

$$\begin{aligned} \alpha : w_2 \rightarrow r_2(0) \rightarrow w_0 \rightarrow r_0(0) \rightarrow r_0(1) \rightarrow r_0(2) \rightarrow w_1 \rightarrow \\ \rightarrow r_1(0) \rightarrow r_2(1) \rightarrow r_1(1) \rightarrow r_2(2) \rightarrow r_1(2) \end{aligned}$$

the $view(0, \alpha) = \{0, 2\}$ and $view(1, \alpha) = [2]$ are immediate snapshots, this means the processes 0 and 2 could have read the array instantaneously. In contrast, the $view(2, \alpha) = \{1, 2\}$ does not correspond to a snapshot. For the following consider the process j such that $w_i \rightarrow_\alpha w_j$ for all i .

Proposition 1. *Let α be a wr-execution on I . Then there exist $j \in I$ such that $\text{view}(j, \alpha) = I$.*

Let α be a wr-execution. For $0 \leq k \leq n$, define $\text{Id}_k(\alpha) = \{j \in \text{Id}(\alpha) : |\text{view}(j, \alpha)| = n + 1 - k\}$. An *IS-execution* is a finite sequence $\alpha = \alpha_0, \dots, \alpha_l$ such that α_0 is a wr-execution on $[n]$, and α_{k+1} is a wr-execution on $\text{Id}(\alpha_k) - \text{Id}_k(\alpha_k)$. Given an IS-execution α , Proposition 1 implies $l(\alpha) \leq n + 1$. Moreover $\text{Id}(\alpha_{k+1}) \subseteq \text{Id}(\alpha_k)$ for all $0 \leq k \leq l - 1$. Hence $|\text{Id}(\alpha_k)| \leq n + 1 - k$. Executions α, α' are *equivalent* if $\text{view}(\alpha) = \text{view}(\alpha')$, denoted $\alpha \sim \alpha'$.

Lemma 1. *Let α and α' be IS-executions with $l(\alpha) = l(\alpha')$. Given $0 \leq k \leq l$, (1) If $\alpha \sim \alpha'$ then $\text{Id}(\alpha_k) = \text{Id}(\alpha'_k)$. (2) If $\alpha_k \sim \alpha'_k$ then $\alpha \sim \alpha'$.*

According to the behavior of the protocol in Figure 1, the local view of i is defined as $\text{view}(i, \alpha) = \text{view}(i, \alpha_k)$, if $i \in \text{Id}(\alpha_k) - \text{Id}(\alpha_{k+1})$ and $\text{view}(i, \alpha) = \text{view}(i, \alpha_l)$ for $k = l$. Hence the view of α is defined as $\text{view}(\alpha) = \{(i, \text{view}(i, \alpha)) : i \in [n]\}$.

Lemma 2. *Let $\alpha = \alpha_0, \dots, \alpha_{l+1}$ be an IS-execution, $l(\alpha) = l + 2$. Then $\text{view}(\alpha) = \text{view}(\alpha')$ for some IS-execution α' such that $l(\alpha') = l + 1$.*

The wr-execution $\alpha' = \alpha_0, \dots, \alpha_{l-1}, \alpha'_l$ of the lemma is obtained by, α'_l such that

$$\text{view}(i, \alpha'_l) = \begin{cases} \text{view}(i, \alpha_l), & \text{if } i \in \text{Id}_l(\alpha_l) \\ \text{view}(i, \alpha_{l+1}), & \text{if } i \notin \text{Id}_l(\alpha_l). \end{cases}$$

It follows $\mathcal{E}_l = \{\text{view}(\alpha) : \alpha = \alpha_0, \dots, \alpha_l\}$. Thus, Lemma 2 implies $\mathcal{E}_{l+1} \subseteq \mathcal{E}_l$. For example consider the IS-execution $\alpha = \alpha_0, \alpha_1, \alpha_2$ where

$$\alpha_0 : w_0 \rightarrow r_0(0) \rightarrow r_0(1) \rightarrow r_0(2) \rightarrow w_1 \rightarrow r_1(0) \rightarrow r_1(1) \rightarrow r_1(2) \rightarrow w_2 \rightarrow r_2(0) \rightarrow r_2(1) \rightarrow r_2(2).$$

$$\alpha_1 : w_0 \rightarrow r_0(0) \rightarrow r_0(1) \rightarrow r_0(2) \rightarrow w_1 \rightarrow r_1(0) \rightarrow r_1(1) \rightarrow r_1(2).$$

$$\alpha_2 : w_0 \rightarrow r_0(0) \rightarrow r_0(1) \rightarrow r_0(2).$$

So $\text{view}(\alpha) = \{(0, \{0\}), (1, \{0, 1\}), (2, \{0, 1, 2\})\} \in \mathcal{E}_2 \subseteq \mathcal{E}_1 \subseteq \mathcal{E}_0$, Fig. 2, 3.

3.2 Topological definitions

The following are standard technical definitions, see [18,21]. A (abstract) *simplicial complex* Δ on a finite set V is a collection of subsets of V such that for any $v \in V$, $\{v\} \in \Delta$, and if $\sigma \in \Delta$ and $\tau \subseteq \sigma$ then $\tau \in \Delta$. The elements of V are called *vertices* and the elements of Δ *simplices*. The *dimension* of a simplex σ is $\dim(\sigma) = |\sigma| - 1$. For instance the vertices are 0-simplices. For the purposes of this paper, we adopt the convention that the *void complex* $\Delta = \emptyset$ is a simplicial complex which is different from the *empty complex* $\Delta = \{\emptyset\}$. Given a positive integer n , Δ^n denotes the standard simplicial complex whose vertex set is $[n]$ and every subset of $[n]$ is a simplex. From now on we identify a complex Δ with its collection of subsets. For every simplex τ we denote by $I(\tau)$ the set of all simplices ρ , $\tau \subseteq \rho$. A simplex τ of Δ is called *free* if there exists a maximal simplex σ such that $\tau \subseteq \sigma$ and no other maximal simplex contains τ . The procedure of removing every simplex of $I(\tau)$ from Δ is called a *collapse*.

Let Δ and Γ be simplicial complexes, Δ is *collapsible* to Γ if there exists a sequence of collapses leading from Δ to Γ . The corresponding procedure is denoted by $\Delta \searrow \Gamma$. In particular, if the collapse is elementary with free simplex τ , it is denoted by $\Delta \searrow_{\tau} \Gamma$. If Γ is the void complex, Δ is *collapsible*. The next definition from [21] gives a procedure to collapse a simplicial complex, by collapsing simultaneously by entire orbits of the group action on the vertex set. Let Δ be a simplicial complex with a simplicial action of a finite group G . A simplex τ is called *G-free* if it is free and for all $g \in G$ such that $g(\tau) \neq \tau$, $I(\tau) \cap I(g(\tau)) = \emptyset$. If τ is *G-free*, the procedure of removing every simplex $\rho \in \bigcup_{g \in G} I(g(\tau))$ is called a *G-collapse* of Δ .

Since, if τ is *G-free* then $g(\tau)$ is free as well, the above definition guarantees that all collapses in the orbit of τ can be done in any order *i.e* every *G-collapse* is a collapse. A simplicial complex Δ is *G-collapsible* to Γ if there exist a sequence of *G-collapses* leading from Δ to Γ , it is denoted by $\Delta \searrow_G \Gamma$. In similar way, if the *G-collapse* is elementary with *G-free* simplex τ , the notation $\Delta \searrow_{G(\tau)} \Gamma$ will be used. In the case Γ is the void complex, Δ is called *G-collapsible*. For instance consider a 2-simplex σ , τ a 1-face of σ and the action of \mathbb{Z}_3 over σ , then τ is free but not \mathbb{Z}_3 -free however σ is \mathbb{Z}_3 -collapsible.

3.3 Protocol Complex

Let n be a positive integer. The abstract simplicial complex $WR_l(\Delta^n)$ with $0 \leq l \leq n$ consists of the set of vertices $V = \{(i, view_i) : i \in view_i \subseteq [n]\}$.

A subset $\sigma \subseteq V$ forms a simplex if only if there exist an IS -execution α of length $l + 1$ such that $\sigma \subseteq \text{view}(\alpha)$.

The complex $\text{WR}_0(\Delta^n)$ is called the *protocol complex* of the write/read model and it will be denoted by $\text{WR}(\Delta^n)$. Protocol complexes for the particular cases $n = 1$ and $n = 2$ are shown in Figure 2. In [21] a combinatorial description of the protocol complex View^n associated to the snapshot model is given. There every simplex of View^n is represented as a $2 \times t$ matrix. Every simplex $\sigma \in \text{WR}(\Delta^n)$ can be expressed as

$$W(\sigma) = \begin{pmatrix} V_1 & \cdots & V_{t-1} & [n] \\ I_1 & \cdots & I_{t-1} & I_t \end{pmatrix}$$

where $I_i \cap I_j = \emptyset$ with $i \neq j$ and $I_i \subseteq V_j$ for all $i \leq j$. Moreover we can associate a matrix for every simplex in the complex $\text{WR}_l(\Delta^n)$. This representation implies that $\chi(\Delta^n)$ and View^n are subcomplexes of $\text{WR}(\Delta^n)$. Figure 3 shows the complex $\text{WR}_l(\Delta^2)$. From now on we will write WR_l instead of $\text{WR}_l(\Delta.)$ unless we specify the standard complex. Lemma 2 implies that

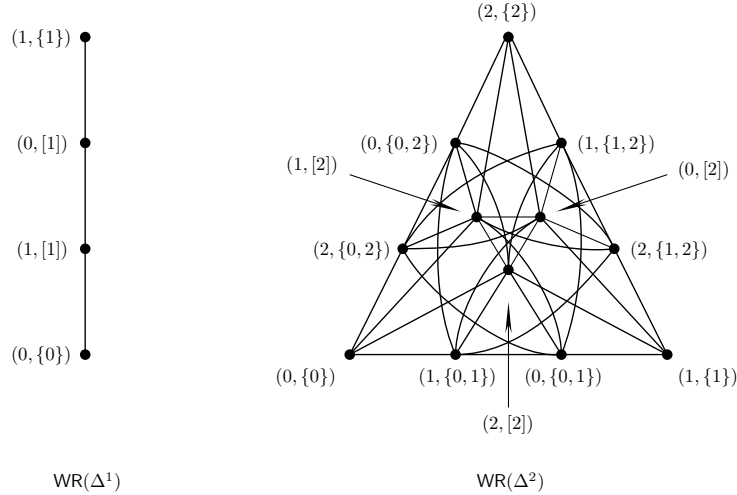


Fig. 2. Protocol Complex for $n = 1$ and $n = 2$.

every maximal simplex of WR_{l+1} is a simplex of WR_l , which implies that WR_{l+1} is a subcomplex of WR_l . From now on σ will denote a simplex of WR_l . For $0 \leq k \leq l$ let $\sigma_k^< = \{(i, \text{view}_i) \in \sigma : |\text{view}_i| < n + 1 - k\}$. In a similar way $\sigma_k^=$ and $\sigma_k = \sigma_k^< \cup \sigma_k^=$ are defined. Therefore, the set of

processes in σ which participate in the $l+1$ call recursive of algorithm 1 is partitioned in those which read $n+1-l$ processes and those which read less than $n+1-l$ processes in the $l+1$ layer shared memory. Let us define $I_\sigma^< = \bigcup_{i \in \text{Id}(\sigma_l^<)} \text{view}_i$ and $I_\sigma = \bigcup_{i \in \text{Id}(\sigma_l)} \text{view}_i$.

Theorem 1. $\sigma \in \text{WR}_{l+1}$ if only if $I_\sigma^< \cap \text{Id}(\sigma_l^=) = \emptyset$ and $|I_\sigma^<| < n+1-l$.

Proof. Suppose $I_\sigma^< \cap \text{Id}(\sigma_l^=) \neq \emptyset$ or $|I_\sigma^<| = n+1-l$. Then there exists a IS -execution $\alpha = \alpha_0, \dots, \alpha_{l+1}$ such that $\sigma_l^< \subseteq \text{view}(\alpha_{l+1})$. In addition there exist processes i and k such that $|\text{view}_i| < n+1-l$, $|\text{view}_k| = n+1-l$ and $k \in \text{view}_i$. This implies that k wrote in the $l+1$ shared memory, a contradiction. For the other direction, since $I_\sigma^< \cap \text{Id}(\sigma_l^=) = \emptyset$ and $|I_\sigma^<| < n+1-l$ we can build an IS -execution $\alpha = \alpha_0, \dots, \alpha_{l+1}$ such that $\sigma \subseteq \text{view}(\alpha)$. \square

Notice that I_σ represents the set of processes which have been read in the $l+1$ recursive call of the algorithm in Figure 1.

Corollary 1. If $\sigma \notin \text{WR}_{l+1}$ then

1. $|I_\sigma| = n+1-l$.
2. $I_\sigma = I_\tau$ for all $\sigma \subseteq \tau$.

Let $\text{inv}(\sigma) = \{(k, I_\sigma) : k \in I_\sigma \setminus I_\sigma^<\}$ if $I_\sigma^< \neq I_\sigma$ else $\text{inv}(\sigma) = \{(k, I_\sigma) : k \in I_\sigma \setminus \text{Id}(\sigma_l^<)\}$. Notice that if $\sigma \notin \text{WR}_{l+1}$ then $\text{inv}(\sigma) \neq \emptyset$.

For the simplices $\sigma^- = \sigma - \text{inv}(\sigma)$ and $\sigma^+ = \sigma \cup \text{inv}(\sigma)$.

Proposition 2. If $\sigma \notin \text{WR}_{l+1}$ then

1. $\sigma^+ = \sigma^- \cup \text{inv}(\sigma)$.
2. $\sigma^- \subseteq \sigma \subseteq \sigma^+$.
3. $\sigma^- \notin \text{WR}_{l+1}$.
4. If $\sigma^- \subseteq \tau \subseteq \sigma^+$ then $\sigma^- = \tau^-$ and $\sigma^+ = \tau^+$.
5. $(\sigma^-)^- = \sigma^-$.

Consider $I_-^+(\sigma) = \{\tau \in \text{WR}_l : \sigma^- \subseteq \tau \subseteq \sigma^+\}$. Item (3) above implies that $I_-^+(\sigma) \cap \text{WR}_{l+1} = \emptyset$ if $\sigma \notin \text{WR}_{l+1}$. Moreover from (4) it is obtained that $I_-^+(\sigma) \cap I_-^+(\tau) = \emptyset$ or $I_-^+(\sigma) = I_-^+(\tau)$.

4 Collapsibility

Let $S_{[n]}$ denote the permutation group of $[n]$. Notice that if the Id 's of processes in a wr -execution on I are permuted according to $\pi \in S_{[n]}$ then we obtain a new linear order on $\pi(I)$. In other words if α is a wr -execution

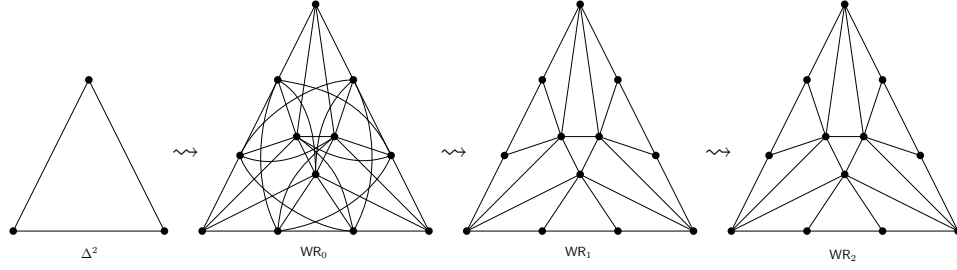


Fig. 3. Complexes WR_l .

on I and $\pi \in S_{[n]}$ then $\alpha' = \pi(\alpha)$ is a *wr*-execution on $\pi(I)$. Moreover if $\sigma = \text{view}(\alpha)$ then $\pi(\sigma) = \text{view}(\pi(\alpha))$. This shows that there exists a natural group action on each simplicial complex WR_l .

Proposition 3. *Let $\sigma \in WR_l$ be a simplex. Then*

1. $\pi(\sigma) \in WR_l$.
2. $\pi(\sigma^-) = \pi(\sigma)^-$.
3. $\pi(\sigma^+) = \pi(\sigma)^+$.
4. $\pi(I_-^+(\sigma)) = I_-^+(\pi(\sigma))$.

For example in Fig. 2, $\sigma = \{(1, \{1, 2\}), (2, \{0, 2\}), (0, [2])\}$ and $\pi(0) = 1$, $\pi(1) = 2$ and $\pi(2) = 0$, then $\pi(\sigma) = \{(2, \{0, 2\}), (0, \{0, 1\}), (1, [2])\}$.

Theorem 2. *For every $0 \leq l \leq n + 1$,*

1. WR_l is collapsible to WR_{l+1} .
2. WR_l is $S_{[n]}$ -collapsible to WR_{l+1} .

Proof. Since $\sigma \in I_-^+(\sigma)$ for all simplices $\sigma \in WR_l$, the intervals $I_-^+(\sigma)$ cover $L = \{\sigma : \sigma \in WR_l, \sigma \notin WR_{l+1}\}$. Additionally, Proposition 2 (4) implies that L can be decomposed as a disjoint union of intervals

$$I_-^+(\sigma_1), \dots, I_-^+(\sigma_k)$$

such that $\sigma_i = \sigma_i^+$ for all $1 \leq i \leq k$. Suppose $\dim(\sigma_{i+1})_l^< \leq \dim(\sigma_i)_l^<$ or if $\dim(\sigma_{i+1})_l^< = \dim(\sigma_i)_l^<$ then $\dim(\sigma_{i+1}) \leq \dim(\sigma_i)$. We will prove by induction on i , $1 \leq i \leq k$, that

$$WR_l^i \searrow_{\sigma_i^-} WR_l^{i+1}$$

where $WR_l^1 = WR_l$ and $WR_l^{k+1} = WR_{l+1}$. If there exists a maximal simplex $\sigma \in WR_l^i$ such that $\sigma_i \subseteq \sigma$ then $\sigma = \sigma_j$ for some $i \leq j \leq k$. Hence

$(\sigma_i)_l^< \subseteq (\sigma_j)_l^<$ and therefore $\sigma_i = \sigma_j$. Now suppose there exists a maximal simplex $\sigma_j \in \text{WR}_l^i$ with $i \leq j \leq k$ such that $\sigma_i^- \subseteq \sigma_j$. This implies that $(\sigma_i)_l^< = (\sigma_j)_l^<$ and $\text{inv}(\sigma_i) = \text{inv}(\sigma_j)$. Thus $\sigma_i = \sigma_i^- \cup \text{inv}(\sigma_i) \subseteq \sigma_j \cup \text{inv}(\sigma_j) = \sigma_j$ and therefore σ_i^- is free in WR_l^i . Therefore,

$$\text{WR}_l = \text{WR}_l^1 \searrow_{\sigma_1^-} \dots \searrow_{\sigma_k^-} \text{WR}_l^{k+1} = \text{WR}_{l+1}.$$

Now if we specify in more detail the order of the sequence, the complex WR_l can be collapsed to WR_{l+1} in a $S_{[n]}$ -equivariant way. First note that if $\pi(\sigma_i) \in \mathbf{I}_+^+(\sigma_j)$ for some $1 \leq j \leq k$, then Proposition 3 (3) and Proposition 2 (4) imply that $\pi(\sigma_i) = \sigma_j$. Moreover, $\dim(\sigma_i)_l^< = \dim \pi(\sigma_i)_l^<$ and $\dim(\sigma_i) = \dim \pi(\sigma_i)$. Hence the set $\{\sigma_1, \dots, \sigma_k\}$ can be partitioned according to the equivalence relation given by: $\sigma_i \sim \sigma_j$ if there exists $\pi \in S_{[n]}$ such that $\pi(\sigma_i) = \sigma_j$. Let τ_1, \dots, τ_p be representatives of the equivalence classes which satisfy the order given in the proof of the item 1, then

$$\text{WR}_l \searrow_{S_{[n]}(\tau_1^-)} \dots \searrow_{S_{[n]}(\tau_p^-)} \text{WR}_{l+1}.$$

□

Fig. 4 illustrates the collapsing procedure $\text{WR}_0 \searrow_{S_{[n]}} \text{WR}_1$ for $n = 2$. In this case consider the simplexes $\sigma_1 = \{(1, \{1, 2\}), (2, \{0, 2\}), (0, [2])\}$ and $\sigma_2 = \{(0, \{0, 1\}), (1, [2]), (2, [2])\}$ then

$$\text{WR}_0^1 \searrow_{S_{[n]}(\sigma_1^-)} \text{WR}_0^2 \searrow_{S_{[n]}(\sigma_2^-)} \text{WR}_0^3 = \text{WR}_1$$

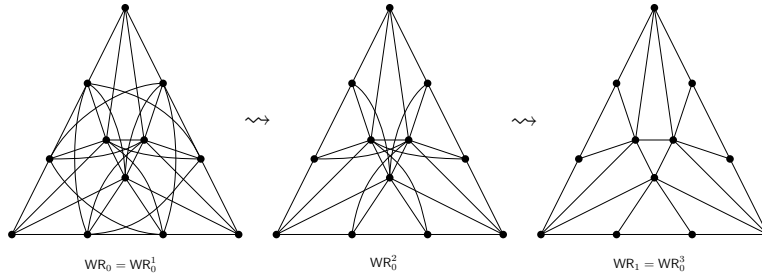


Fig. 4. $S_{[n]}$ -collapse.

We have the following consequence.

Corollary 2. *For every natural number n , the simplicial complex $\text{WR}(\Delta^n)$ is $S_{[n]}$ -collapsible to $\chi(\Delta^n)$.*

Multi-round protocol complex A *carrier map* Φ from complex \mathcal{C} to complex \mathcal{D} assigns to each simplex σ a subcomplex $\Phi(\sigma)$ of \mathcal{D} such that $\Phi(\tau) \subseteq \Phi(\sigma)$ if $\tau \subseteq \sigma$. The protocol complex of the iterated write/read model (see Fig. 5), $k \geq 0$, is $WR^{(k+1)}(\Delta^n) = \bigcup_{\sigma \in WR^{(k)}(\Delta^n)} WR(\sigma)$.

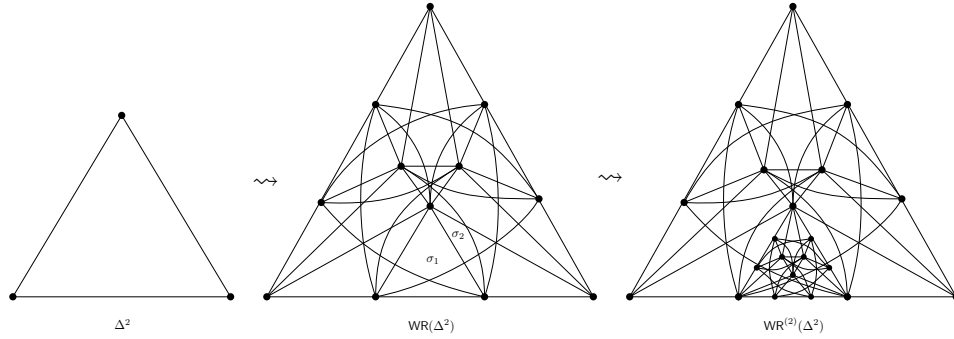


Fig. 5. Complexes of the iterated model; in $WR^{(2)}(\Delta^2)$ only $WR(\sigma_1)$ is depicted.

Corollary 3. For all $k \geq 1$, $WR^{(k)}(\Delta^n) \searrow \chi^{(k)}(\Delta^n)$.

The collapsing procedure consists first in collapsing, in parallel, each subcomplex $WR(\sigma)$ where σ is a maximal simplex of $WR^{(k-1)}(\Delta^n)$ as in Theorem 2. An illustration is in Fig. 6, applied to the simplexes $\sigma_1 = \{(0, \{0, 1\}), (1, \{0, 1\}), (2, [2])\}$ and $\sigma_2 = \{(0, \{0, 1\}), (1, [2]), (2, [2])\}$ of $WR(\Delta^2)$. Second, we collapse $\chi(WR^{(k-1)}(\Delta^n))$ to $\chi^{(k)}(\Delta^n)$. For example σ_2 is collapsed to $\tau = \{(0, \{0, 1\}), (1, [2])\}$, see Fig. 7, then Proposition 4 implies that $\chi(\sigma_1)$ is collapsed to $\chi(\tau)$.

References

1. Yehuda Afek, Hagit Attiya, Danny Dolev, Eli Gafni, Michael Merritt, and Nir Shavit. Atomic snapshots of shared memory. *J. ACM*, 40(4):873–890, Sep. 1993.
2. Hagit Attiya and Sergio Rajsbaum. The combinatorial structure of wait-free solvable tasks. *SIAM J. Comput.*, 31(4):1286–1313, April 2002.
3. Ofer Biran, Shlomo Moran, and Shmuel Zaks. A Combinatorial Characterization of the Distributed 1-Solvable Tasks. *J. Algorithms*, 11(3):420–440, 1990.
4. Elizabeth Borowsky and Eli Gafni. Generalized flip impossibility result for t -resilient asynchronous computations. In *Proc. 25th Annual ACM Symp. on Theory of Computing*, STOC, pages 91–100, New York, NY, USA, 1993. ACM.

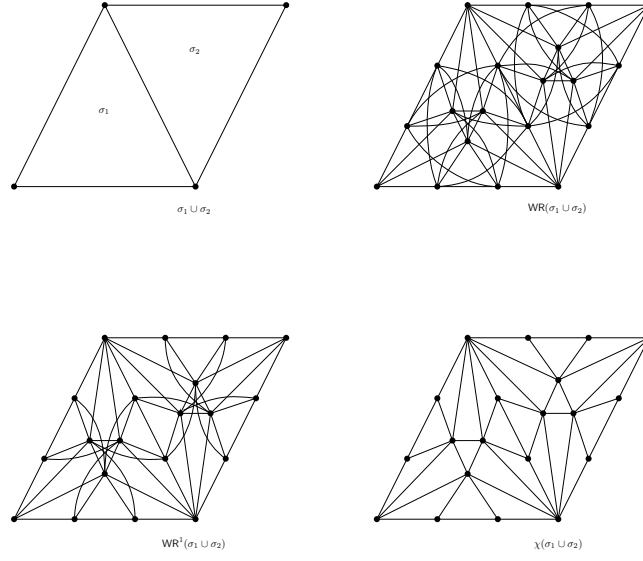


Fig. 6. First Collapsing.

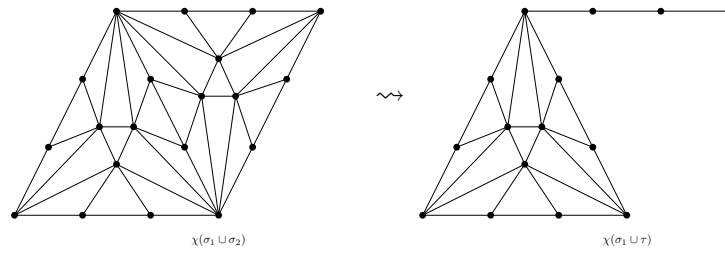


Fig. 7. Second Collapsing.

5. Elizabeth Borowsky and Eli Gafni. Immediate atomic snapshots and fast renaming. In *Proc. 12th ACM Symp. on Principles of Distributed Computing*, PODC, pages 41–51, New York, NY, USA, 1993. ACM.
6. Elizabeth Borowsky and Eli Gafni. A simple algorithmically reasoned characterization of wait-free computation (extended abstract). In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '97, pages 189–198, New York, NY, USA, 1997. ACM.
7. Elizabeth Borowsky, Eli Gafni, Nancy Lynch, and Sergio Rajsbaum. The BG distributed simulation algorithm. *Distributed Computing*, 14(3):127–146, 2001.
8. M. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility Of Distributed Commit With One Faulty Process. *Journal of the ACM*, 32(2), April 1985.
9. Eli Gafni and Sergio Rajsbaum. Recursion in distributed computing. In Shlomi Dolev, Jorge Cobb, Michael Fischer, and Moti Yung, editors, *Stabilization, Safety, and Security of Distributed Systems*, volume 6366 of *Lecture Notes in Computer Science*, pages 362–376. Springer Berlin Heidelberg, 2010.
10. Eric Goubault, Samuel Mimram, and Christine Tasson. Iterated chromatic subdivisions are collapsible. *Applied Categorical Structures*, pages 1–42, 2014.
11. John Havlicek. Computable obstructions to wait-free computability. *Distributed Computing*, 13(2):59–83, 2000.
12. John Havlicek. A note on the homotopy type of wait-free atomic snapshot protocol complexes. *SIAM J. Comput.*, 33(5):1215–1222, 2004.
13. Maurice Herlihy, Dmitry Kozlov, and Sergio Rajsbaum. *Distributed Computing Through Combinatorial Topology*. Elsevier, Imprint Morgan Kaufmann, 2013.
14. Maurice Herlihy and Sergio Rajsbaum. Simulations and reductions for colorless tasks. In *Proceedings of the 2012 ACM symposium on Principles of distributed computing*, PODC '12, pages 253–260, New York, NY, USA, 2012. ACM.
15. Maurice Herlihy and Nir Shavit. The topological structure of asynchronous computability. *J. ACM*, 46(6):858–923, November 1999.
16. Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.
17. Gunnar Hoest and Nir Shavit. Towards a topological characterization of asynchronous complexity. In *Proc. 16th ACM Symp. Principles of distributed computing*, PODC, pages 199–208, New York, NY, USA, 1997. ACM.
18. Jakob Jonsson. *Simplicial Complexes of Graphs*. 10.1007/978-3-540-75859-4. Lecture Notes in Mathematics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
19. Dmitry N. Kozlov. Chromatic subdivision of a simplicial complex. *Homology Homotopy Appl.*, 14(2):197–209, 2012.
20. Dmitry N. Kozlov. Topology of the immediate snapshot complexes. *Topology Appl.*, 178:160–184, 2014.
21. Dmitry N. Kozlov. Topology of the view complex. *Homology Homotopy Appl.*, 17(1):307–319, 2015.
22. M. C. Loui and H. H. Abu-Amara. *Memory requirements for agreement among unreliable asynchronous processes*, volume 4, pages 163–183. JAI press, 1987.
23. Sergio Rajsbaum, Michel Raynal, and Corentin Travers. The iterated restricted immediate snapshot model. In *COCOON*, volume 5092 of *Lecture Notes in Computer Science*, pages 487–497. Springer, 2008.
24. Michael Saks and Fotios Zaharoglou. Wait-Free k-Set Agreement is Impossible: The Topology of Public Knowledge. *SIAM J. Comput.*, 29(5):1449–1483, 2000.

Appendix

A Proofs

Lemma 1 Let α and α' be *IS*-executions with $l(\alpha) = l(\alpha')$. Given $0 \leq k \leq l$,

1. If $\alpha \sim \alpha'$ then $\text{Id}(\alpha_k) = \text{Id}(\alpha'_k)$.
2. If $\alpha_k \sim \alpha'_k$ then $\alpha \sim \alpha'$.

Proof. To see 1 note that $\text{Id}_k(\alpha_k) = \text{Id}_k(\alpha'_k)$ since $\text{view}(\alpha) = \text{view}(\alpha')$. Hence $\text{Id}(\alpha_k) = \text{Id}(\alpha'_k)$. Now, 2 is a consequence of equality $\text{view}(\alpha_k) = \text{view}(\alpha'_k)$. \square

Proposition 2 If $\sigma \notin \text{WR}_{l+1}$ then

1. $\sigma^+ = \sigma^- \cup \text{inv}(\sigma)$.
2. $\sigma^- \subseteq \sigma \subseteq \sigma^+$.
3. $\sigma^- \notin \text{WR}_{l+1}$.
4. If $\sigma^- \subseteq \tau \subseteq \sigma^+$ then $\sigma^- = \tau^-$ and $\sigma^+ = \tau^+$.
5. $(\sigma^-)^- = \sigma^-$.

Proof. Items 1 and 2 follow directly from the definitions of σ^- and σ^+ . Now, for 3, notice that $\sigma_l^< = (\sigma^-)_l^<$, which implies $\text{I}_\sigma^< = \text{I}_{\sigma^-}^<$. Therefore the statement follows from Theorem 1. To prove 4, first note that $\tau = \sigma^- \cup \rho$, where $\rho \subseteq \text{inv}(\sigma)$. Therefore $\sigma_l^< = \tau_l^<$ and $\text{I}_\sigma = \text{I}_\tau$ so $\text{inv}(\sigma) = \text{inv}(\tau)$. This implies that

$$\sigma^- = (\sigma - \text{inv}(\sigma)) - \text{inv}(\sigma) \subseteq \tau - \text{inv}(\tau) \subseteq (\sigma \cup \text{inv}(\sigma)) - \text{inv}(\sigma) = \sigma^-$$

then $\sigma^- = \tau^-$. From 1 it is obtained

$$\tau^+ = \tau^- \cup \text{inv}(\tau) = \sigma^- \cup \text{inv}(\sigma) = \sigma^+.$$

Finally, to prove (5), observe that $(\sigma^-)_l^< = \sigma_l^<$ and $\text{I}_{\sigma^-} = \text{I}_\sigma$. Thus $\text{inv}(\sigma^-) = \text{inv}(\sigma)$, which proves

$$(\sigma^-)^- = \sigma^- - \text{inv}(\sigma^-) = (\sigma - \text{inv}(\sigma)) - \text{inv}(\sigma) = \sigma - \text{inv}(\sigma) = \sigma^-.$$

\square

Proposition 3 Let $\sigma \in \text{WR}_l$ be a simplex.

1. $\pi(\sigma) \in \text{WR}_l$.

2. $\pi(\sigma^-) = \pi(\sigma)^-$.
3. $\pi(\sigma^+) = \pi(\sigma)^+$.
4. $\pi(I_-^+(\sigma)) = I_-^+(\pi(\sigma))$.

Proof. Item (1) is clear. (2) and (3) are deduced from the equalities $\pi(\sigma_l^<) = \pi(\sigma)_l^<$ and $\pi(\sigma_l^-) = \pi(\sigma)_l^-$. The last item follows from Proposition 2 (4). \square

Let Δ, Γ be simplicial complexes. Consider the set $C = \{\tau_1, \dots, \tau_k\}$ of free simplexes in Δ such that $I(\tau_i) \cap I(\tau_j) = \emptyset$ for all $i \neq j$. This equation means that we can collapse τ_1, \dots, τ_k in a parallel way. This will be denoted by $\Delta \searrow_C \Gamma$.

Corollary 3 For all $k \geq 1$, $\text{WR}^{(k)}(\Delta^n) \searrow \chi^{(k)}(\Delta^n)$.

Proof. The proof is by induction on k . Theorem 2 proves the base case $k = 1$. For $k > 1$, let $\sigma_1, \dots, \sigma_m$ be the maximal simplexes of $\text{WR}^{(k-1)}(\Delta^n)$. Now consider σ_1 as a standard simplicial complex, then let $\tau_1^-, \dots, \tau_k^-$ be a sequence of the $S_{[n]}$ -collapsability

$$\text{WR}_l(\sigma_1) \searrow_{S_{[n]}} \text{WR}_{l+1}(\sigma_1)$$

with $0 \leq l \leq n-1$ where $\text{WR}_0(\sigma_1) = \text{WR}(\sigma_1)$ and $\text{WR}_n(\sigma_1) = \chi(\sigma_1)$. On the other hand there exists a isomorphism f_i from $\text{WR}(\sigma_1)$ to $\text{WR}(\sigma_i)$ for all $1 \leq i \leq m$. This implies that $f_i(\tau_1^-), \dots, f_i(\tau_k^-)$ is a sequence of the $S_{[n]}$ -collapsability

$$\text{WR}_l(\sigma_i) \searrow_{S_{[n]}} \text{WR}_{l+1}(\sigma_i)$$

for all $1 \leq i \leq m$. Consider $G = \{f_1, \dots, f_m\}$ with f_1 the identity simplicial map and $C_j = S_{[n]}(G(\tau_j^-))$ with $1 \leq j \leq k$. Then for all $0 \leq l \leq n-1$

$$\text{WR}_l^{(k)}(\Delta^n) \searrow_{C_1} \dots \searrow_{C_k} \text{WR}_{l+1}^{(k)}(\Delta^n)$$

where $\text{WR}_0^{(k)}(\Delta^n) = \text{WR}^{(k)}(\Delta^n)$ and $\text{WR}_n^{(k)}(\Delta^n) = \chi(\text{WR}^{(k-1)}(\Delta^n))$. Thus

$$\text{WR}^{(k)}(\Delta^n) \searrow \chi(\text{WR}^{(k-1)}(\Delta^n)).$$

By induction hypothesis

$$\text{WR}^{(k-1)}(\Delta^n) \searrow \chi^{(k-1)}(\Delta^n).$$

Then proposition 4 implies that

$$\chi(\text{WR}^{(k-1)}(\Delta^n)) \searrow \chi(\chi^{(k-1)}(\Delta^n))$$

and therefore

$$\text{WR}^{(k)}(\Delta^n) \searrow \chi^{(k)}(\Delta^n).$$

\square

B Chromatic Subdivision

In the proof of Corollary 3 we used Proposition 4 which appears in [10]. For completeness, we present the proof in our notation.

Let σ be a simplex of $\chi(\Delta^n)$. Notice there exists a vertex $(i, view_i) \in \sigma_0^<$ such that $view_j \subseteq view_i$ for all $(j, view_j) \in \sigma_0^<$. Then σ^c will denote the simplex

$$\sigma^c = \sigma \cup \{(j, [n]) : j \in [n] \setminus view_i\}.$$

Lemma 3. *Let n be a natural number. Then $\chi(\Delta^n)$ is collapsible.*

Proof. For $1 \leq k \leq n+1$ let

$$\Delta_k^n = \Delta_{k-1}^n \setminus \bigcup_{\sigma \in \Sigma_{k-1}} I(\sigma_0^<)$$

with $\Delta_0^n = \chi(\Delta^n)$ and $\Sigma_{k-1} = \{\sigma \in \Delta_{k-1}^n : \dim \sigma_0^< = n - k\}$. Then the simplex $\sigma_0^<$ is a free face of σ^c in the simplicial complex Δ_{k-1}^n . Hence

$$\chi(\Delta^n) = \Delta_0^n \searrow \cdots \searrow \Delta_{n+1}^n$$

where Δ_{n+1}^n is the void complex. □

For each natural number n and $p \in [n]$ let

$$\Lambda_p^n = \Delta^n \setminus I(\sigma')$$

where $\sigma' = [n] \setminus \{p\}$.

Proposition 4. *For all natural numbers n*

$$\chi(\Delta^n) \searrow \chi(\Lambda_p^n).$$

Proof. The collapsing procedure consists in three stages.

Stage 1. Let

$$\Sigma = \{\sigma \in \chi(\Delta^n) : \sigma_0^= = \{(p, [n])\}\}.$$

Given $1 \leq k \leq n$, consider the simplicial complex

$$K_k^n = K_{k-1}^n \setminus \bigcup_{\sigma \in \Sigma_{k-1}} I(\sigma_0^<)$$

with $K_0^n = \chi(\Delta^n)$ and $\Sigma_{k-1} = \{\sigma \in \Sigma \cap K_{k-1}^n : \sigma \text{ is maximal}\}$. Notice $\sigma_0^<$ is a free face of σ since $\dim \sigma = \dim \sigma_0^< + 1$. Therefore

$$\chi(\Delta^n) = K_0^n \searrow \cdots \searrow K_n^n.$$

Stage 2. Now consider the simplexes

$$\Sigma' = \{\sigma \in K_n^n : (p, [n]) \in \sigma\}.$$

Given $1 \leq k \leq n$ let

$$T_k^n = T_{k-1}^n \setminus \bigcup_{\sigma \in \Sigma'_{k-1}} I(\sigma|p)$$

where $\Sigma'_{k-1} = \{\sigma \in \Sigma' \cap T_{k-1}^n : \dim \sigma_0^< = n - k - 1\}$ and $\sigma|p = \sigma_0^< \cup \{(p, [n])\}$. Then $\sigma|p$ is a free face of σ^c with $\sigma \in \Sigma'$. Hence,

$$K_n^n = T_0^n \searrow \cdots \searrow T_n^n.$$

Stage 3. For k , $1 \leq k \leq n$, let

$$M_k = M_{k-1} \setminus \bigcup_{\sigma \in \Sigma''_{k-1}} I(\sigma_0^<)$$

where $M_0 = T_n^n$ and $\Sigma''_{k-1} = \{\sigma \in K_n^n : \dim \sigma_0^< = n - k\}$. Now suppose $\sigma_1, \dots, \sigma_l$ are the simplexes such that $\sigma = (\sigma_i)_0^<$ and $\dim(\sigma_i)_0^< = n - k$. Notice if $\sigma \in \Sigma''_{k-1}$ then there exist simplexes τ and σ_i such that $\sigma = \tau \cup \sigma_i$ with $\tau \in \chi(\Delta^{I_i})$ and $I_i = [n] \setminus \text{Id}(\sigma_i)$. For each i , $1 \leq i \leq l$, let $\tau_1^{(i)}, \dots, \tau_m^{(i)}$ be the sequence of collapses of $\chi(\Delta^{I_i})$ given in the Lemma 3. Therefore

$$\tau_1^{(1)} \cup \sigma_1, \dots, \tau_m^{(1)} \cup \sigma_1, \dots, \tau_1^{(l)} \cup \sigma_l, \dots, \tau_m^{(l)} \cup \sigma_l$$

is a sequence of collapses leading from M_{k-1} to M_k . Hence

$$M_0 \searrow \cdots \searrow M_n = \chi(\Lambda_p^n).$$

□